**Claim Listing**

1. (Previously presented) An asymmetric data processor comprising:

a first group of nodes comprising one or more host processors, each host comprising a memory, a network interface, and one or more Central Processing Units (CPUs), wherein each host accepts and responds to queries for data, and transforms such queries into one or more jobs;

a second group of nodes comprising a plurality of Job Processing Units (JPUs), wherein each JPU comprises:

a memory, for storing data;

a network interface, for receiving data and instructions;

a streaming data interface, for receiving data from a streaming data source;

one or more general purpose CPUs, for responding to requests from at least one host computer in the first group, and to requests from other JPUs in the second group; and

one or more Programmable Streaming Data Processors (PSDPs) configured to perform filtering functions directly on data received from the streaming data interface, each PSDP thus performing initial processing on a set of data; and

a network connecting the nodes within each group and between the two groups; and

wherein a JPU at the second group of nodes is configured to receive jobs from one or more nodes in the first group, perform work requested by the jobs, and generate a result based on the work.

2. (Original) The apparatus of claim 1 wherein the data comprises structured records, and the structured records further comprise fields of various lengths and data types.

3. (Previously presented) An apparatus as in claim 1 wherein the filtering functions performed by the PSDPs comprise field-level filtering.

4. (Original) An apparatus as in claim 1 wherein the streaming data interface is an industry standard mass storage interface.

5. (Original) An apparatus as in claim 2 in which at least one selected PSDP performs Boolean comparisons of record field values against other values.

6. (Previously presented) An apparatus as in claim 5 wherein the Boolean comparison is against at least one of other record field values and values held internally to that PSDP.

7. (Original) An apparatus as in claim 5 in which the selected PSDP restricts records that fail Boolean comparisons of field values, as such records stream into the PSDP and before such records are placed into the memory of the associated JPU.

8. (Original) An apparatus as in claim 2 in which the selected PSDP filters out fields of records that are not needed for particular queries, as such fields stream into the PSDP and before such fields are placed into the memory of the associated JPU, projecting forward into JPU memory those fields that are needed.

9. (Original) An apparatus as in claim 2 in which the PSDP output data may contain projected fields not contained in the source data, such as row address, transforms, results of expression evaluation, results of bit joins, and results of visibility tests.

10. (Previously presented) An apparatus as in claim 2 in which a selected PSDP decompresses at least one of fields and records.

11. (Original) An apparatus as in claim 1 wherein the streaming data interface is connected to receive data from a peripheral device selected from the group consisting of disk drive, network interface, and other streaming data source.

12. (Original) An apparatus as in claim 2 in which a selected PSDP performs a join operation, where the field values being joined have a small range of values, so that the presence or absence of a particular value can then be encoded as a bit within a sequence of bits, whose position within the sequence corresponds to the field value.

13. (Original) An apparatus as in claim 2 in which a selected PSDP performs an "exist join" operation, where the field values being joined have a small range of values, so that the presence or absence of a particular value can then be encoded as a bit within a sequence of bits, whose position within the sequence corresponds to the field value.

14. (Original) An apparatus as in claim 1 in which space is reserved in JPU memory at the head of the first tuple produced by the PSDP for recording tuple length and null vector, so that the length and null vectors from the end of the tuple may be relocated to this space.

15. (Original) An apparatus as in claim 1 in which at least one PSDP is implemented as a Field Programmable Gate Array (FPGA).

16. (Previously presented) An apparatus as in claim 1 in which the host computers in the first group contain software comprising a plan optimizer component that determines which filtering functions should be executed within a PSDP.

17. (Previously presented) An apparatus as in claim 1 in which the JPUs in the second group contain software comprising a plan optimizer component that determines which filtering functions should be executed within a PSDP.

18. (Original) An apparatus as in claim 1 in which the host computers in the first group contain software comprising a plan link component, which determines a query execution plan, the query execution plan further having portions that will be processed by a PSDP, portions that will be processed by a JPU after a PSDP has returned data to the JPU, and

portions that will be processed by a host, after the JPU has returned data to the host group.

19. (Original) An apparatus as in claim 1 in which the JPUs in the second group contain software comprising a plan link component, which determines a query execution plan, the query execution plan further having portions that will be processed by a PSDP, portions that will be processed by a JPU after a PSDP has returned data to the JPU, and portions that will be processed by a host, after the JPU has returned data to the host group.

20. (Previously presented) An apparatus as in claim 1 in which the hosts in the first group contain software comprising a PSDPPrep component, which, for a given query execution plan, defines filtering instructions.

21. (Previously presented) An apparatus as in claim 1 in which the JPUs in the second group contain software comprising a PSDPPrep component, which, for a given query execution plan, defines filtering instructions.

22. (Original) An apparatus as in claim 21 wherein the instructions defined by the PSDPPrep component include instructions to process fields of records.

23. (Previously presented) An apparatus as in claim 21 in which a PSDPPrep component further identifies at least one of filtering, transformation, projection and aggregation operations to be performed by a PSDP.

24. (Original) An apparatus as in claim 21 in which a PSDPPrep component further modifies the query execution plan to specify restrict operations that are to be performed by a PSDP instead of a JPU.

25. (Original) An apparatus as in claim 1 in which the JPUs contain software comprising a PSDP Filter component, which loads an executable code image into a PSDP.

26. (Original) An apparatus as in claim 1 in which the JPUs contain software comprising a PSDP Scheduler component, which schedules jobs to run on a PSDP and queues PSDP requests to retrieve required data.

27. (Original) An apparatus as in claim 1 in which the JPUs in the second group contain software comprising a JPU Resource Scheduler component, which is responsible for scheduling jobs to be run on the JPU.

28. (Original) An apparatus as in claim 27 in which the JPU Resource Scheduler component further schedules jobs to run on a PSDP, communicating with a PSDP Scheduler component to queue up PSDP requests to retrieve required data.

29. (Original) An apparatus as in claim 27 in which the JPU Resource Scheduler component further schedules jobs, in which similar PSDP instructions in different query execution plans are combined to avoid duplicate PSDP processing requests.

30. (Previously presented) An apparatus as in claim 2 in which an initial query is provided by a structured query language (SQL) statement, and the records specified thereby exist in various processing states within at least two components of the system, the two components including at least one of a PSDP within a JPU and a host.

31. (Original) An apparatus as in claim 30 in which a PSDP processes fields within records are received from the streaming data source, without waiting to process any records until all records are received.